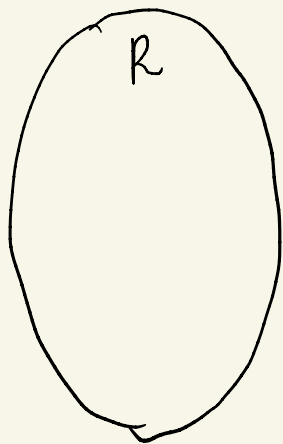


The study of computable functions



Example. The Fibonacci Number.

$$\forall n \in \mathbb{N} \quad F(n) = \begin{cases} F(n-1) + F(n-2), & n \geq 2 \\ 1, & n = 1 \\ 0, & n = 0 \end{cases}$$

Fib1(n)

if $n=0$: return 0;

if $n=1$: return 1;

return Fib1(n-1) + Fib1(n-2)

The "running time" $\rightarrow$ # of "steps"

$$T(n) \leq 2 \quad \text{for } n \leq 1$$

$$T(n) = T(n-1) + T(n-2) + 3. \quad \text{for } n > 1.$$

$$T(n) \gg F(n) = 2^{O(n)}.$$

Fib2(n)

```
int f[n];
```

```
f[0] = 0; f[1] = 1;
```

```
for i = 2, ..., n:
```

```
    f[i] = f[i-1] + f[i-2];
```

```
return f[n]
```

$$T(n) = O(n) ?$$

" Can we treat $f(i) = f(i-1) + f(i-2)$ an atomic step.

$F(n) = 2^{O(n)}$ - a $O(n)$ bits long string.

Adding two n -length strings costs $O(n)$.

Therefore $T(n) = O(n^2)$.

$$\begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} F_{n-1} \\ F_n \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}.$$

How to compute $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n$?

$O(\log n)$ multiplication operations

Each operation: multiply two $O(n)$ strings, costs $M(n)$

$$O(M(n) \cdot \log n)$$

$$M(n) \geq n.$$

Careful Analysis

$$M(1) + M(2) + M(4) + M(8) + \dots + M(n)$$

Prop: $\sum_{i=0}^K M(2^i) \leq 2M(2^K).$

Induct: $\sum_{i=0}^{K-1} M(2^i) + M(2^K) \leq 2M(2^{K-1}) + M(2^K) \leq M(2^K).$

* What is $M(n)$?

$$M(n) = n^2 \dots$$

$$M(n) = n \log n \quad \text{FFT.}$$

Fib4 : Direct evaluation ?

$$F_n = \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2} \right)^n - \frac{1}{\sqrt{5}} \left(\frac{1-\sqrt{5}}{2} \right)^n$$

not possible in our model.

View the input of F_n as unary number.

Fib2, Fib3 are polynomial-time algo.

the notion for efficient algorithms.

Why polynomial-time ?

① grow much slower than exponential.

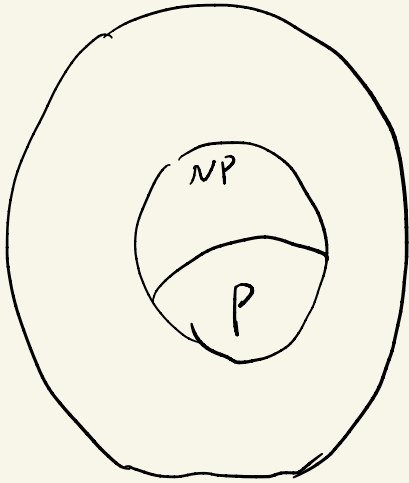
② closed under composition

$$A(B(C(x)))$$

③ Robust to machine model.

complexity-theoretic Church Turing

Inside R.



$$NP = f: \{0,1\}^* \rightarrow \{0,1\}$$

efficiently verifiable:

$\exists$ polytime TM T . $\exists$ poly P

$$\forall x. f(x)=1 \Leftrightarrow \exists y \in \{0,1\}^{P(|x|)}.$$

$$T(x, y) = 1.$$

$$NP \neq P.$$